The University of Kansas
Mechanical Engineering Department

Project Final Report

Powered Wheelchair

Roll Over Prevention System

Team 4
Emma Bartelsen, Hannah Chrisman,
Alex Estes, Mathew Nervaiz
ME 643 - Capstone Design Project
Course Instructor - Dr. Robert Sorem
Advisors - Dr. Kenneth Fischer and Dr. Sara Wilson
Sponsor - Executive Director at MS Achievement Center, Mrs. Judy Markwardt-Oberheu
5.19.22

Executive Summary

The goal of this project was to design and build a sensor system for a power wheelchair with the ability to detect upcoming obstacles capable of tipping the wheelchair and warn a user thus preventing an accident from occurring. Through the course of this project, time was invested into researching the problem, coming up with detailed design solutions in the form of flowcharts, testing sensors, and building a system capable of meeting the desired customer specifications. Customer specifications were determined through thorough research and customer interviews. These specifications included adhering to specific limitations faced by the MS community, not interfering with the basic function of the wheelchair, and limiting any false alarms or other inconveniences to the user. The design process phase was utilized to determine what variables and information were necessary for the project to be completed. At first, it was determined that velocity was a necessary variable as this would allow a warning to be sent to the user at a strategic time determined by their proximity to the obstacle. This variable was later dropped, and the code was simplified to only using distance sensors. The distance sensors used were the TFmini-S Lidar (ToF) laser range sensors. Eight of these sensors were provided for use as well as a Raspberry Pi to run the code on. The Raspberry Pi was later changed to Arduino as there was more familiarity with this technology. Although the Arduino and sensors were easy to use and provided reliable results for code with one sensor, having code that reliably ran more than one sensor proved challenging and ultimately was not achieved. The final design used two separate Arduino boards that ran two separate sensors in the front while the warning system was connected to both boards. One sensor was for close distances while one read further out. This allowed the user more time to react as well as a secondary warning. The further out sensor changed the LED warning system from green to yellow while the close sensor turned on a red LED when danger was approaching. The mounts for the system were designed in SolidWorks and 3D printed. Some key features of the mounts included adjustability, fit to specifications and testing results, and stability of placement. Testing was one of the most important undertakings. Four main types of testing were completed consisting of static distance testing, static angle testing, static surface testing, and dynamic testing. Static distance testing was used to determine the range of accuracy of the sensors. The sensors were determined to be accurate at distances greater than 30 cm to 575 cm, which was the maximum distance tested. This information was used to limit the placement of the sensors to anywhere that would be reading distances greater than 30 cm. Results from static angle testing were very important in the placement of the sensors as it was determined that these sensors are inaccurate at angles greater than 70 degrees. This limited the distance away from the chair that the sensors could read and was a major design factor. Surface testing gave confidence that the sensors could handle grass, concrete, and curved surfaces such as curbs, but didn't return as accurate of results with reflective surfaces like polished tile which can be difficult for these sensors to handle. Dynamic testing was completed after the design was finalized. This testing proved that the project could successfully and reliably warn users of upcoming obstacles. While the overall goal of the project was successful, some modifications and changes could be made to improve for the future. One thing that was missing from this project and that was strongly suggested in customer specifications was a buzzer. This would meet users' needs of having an auditory warning in addition to a visual warning and provide a safer system overall. Another thing that could be improved upon would be getting multiple sensors to run on the same code. Solutions to this issue include getting better sensors, obtaining a more permanent wiring solution, or switching back to Raspberry Pi.

Background and Problem Definition

In an article published by the US National Library of Medicine in 2006, more than 100,000 wheelchair-related injuries were reported annually in the US. Of these incidents, tips and falls accounted for 65 - 80% [1]. Considering that there are 250 thousand Multiple Sclerosis patients in the US that utilize a wheelchair to assist with mobility, the MS community is a significant contributor to these wheelchair incident statistics [2]. MS patients are often unable to properly analyze their surroundings for obstacles due to their mobility limitations. As a result, 57.3% of wheelchair related incidents occurred in an environment with stairs, ramps, or curbs [3]. Therefore, a system with the capability of alerting the user when approaching a drop-off would be a helpful, and potentially life-saving feature. As a result, the project's goal is to create a wheelchair attachable system that alerts the user when approaching an obstacle that puts the user at risk of tipping.

Project Specifications

These specifications were defined by our sponsor and through interviews with patients using powered wheelchairs. The key factors necessary for defining the project specifications are the system's safety and functionality.

The project's specifications include the following:

1. Patients with MS can suffer from the loss of strength and motion in the arms, hands, and neck, as well as loss of vision and sensation in the hands. The system would need to adhere to these limitations and allow the user enough time to redirect the chair after receiving a warning.

2. In addition to adhering to the users' physical limitations, the system also cannot interfere with the wheelchair's basic functions or introduce any new potential hazards. This means that this assistive system should not: shut down the wheelchair, interfere with tilt and recline features, or restrict other common uses like hanging a bag or jacket around the back.

3. Any additions must be properly ventilated, securely wired, sufficiently housed and sealed, and must be able to operate in various weather conditions (below zero to 40 C).

4. False alarms must be limited as much as possible and the system should be able to distinguish between approaching a dangerous obstacle or being within close proximity to the obstacle and whether the obstacle requires immediate attention.

5. The sensor system should be user friendly, easy to attach to a wheelchair, and able to be understood by the user.

Materials

At the beginning of the year, the team was provided with a Raspberry Pi, 8 TFmini-S Lidar (ToF) laser range sensors (Appendix 2H), the necessary wiring, a battery pack, and the power wheelchair. It was expected that we would use this equipment to complete the project. The battery of the powered wheelchair needed to be replaced, however. Due to the expense of purchasing a new battery, the team instead decided to pursue designing a device capable of mounting onto a manual wheelchair provided by Dr. Fischer. The team also decided to switch from the Raspberry Pi to Arduino. Although the Raspberry

Pi provided increased capabilities, it was advised by the course instructor to pursue Arduino due to the team's higher familiarity with the system.

The team spent a majority of the first semester defining how to approach the problem presented to us by our sponsor. This includes defining what variables were needed to accurately detect drop offs and what form of data was necessary to define those variables. This information was organized into multiple rounds of flowcharts. These flowcharts allowed the team to visualize and demonstrate how the code would work, therefore, providing guidance in how to write the code. The primary versions of the flowchart all included velocity and distance variables. These flowcharts planned to utilize velocity to determine if the chair was moving and when to warn the user of an approaching obstacle. As a result, hall effect sensors and accelerometers were purchased to determine velocity. The final version of the flowchart left out the velocity component to simplify the problem by not using multiple sensors at once and as calculating the wheelchair's velocity was not a vital source of data required to alert the user of a drop off. The final design therefore did not incorporate the purchased accelerometers or hall effect sensors. The final flowchart can be seen in Appendix 1A. It outlines a procedure that compares the measured distance to the actual distance and warns the user of any differentiation from the actual distance (detecting an obstacle). It was decided that the time-of-flight sensors should have the capability of collecting the necessary distance measurements. Therefore, new distance sensors were not purchased.

Programming

In terms of written code, the team was successful in running code on Arduino. This allowed the team to collect data utilizing the time-of-flight sensors. The time-of-flight sensors output two measurements: The distance in centimeters and strength of the signal. The strength of the signal ranges from 0-65535 and is dependent on the measured distance and the reflectivity of the object. If the strength of the signal is lower than 100, the time-of-flight sensor classifies the distance measured to be unreliable and outputs a set value of -1 for the distance, which can be classified by the user as an error.

The code was divided into two major milestones. The first was to write a code that permits for more than one time-of-flight sensor to operate synchronously. The second milestone was to implement a statement within the code to indicate a drop off. The team was unable to implement running multiple sensors at once on the Arduino Uno and consulted with Dr. Wilson to discuss troubleshooting the issue. It was decided that it would be unlikely that the Arduino Uno would be able to reliably collect data for more than one sensor at a time. As a result, it was suggested to utilize the Arduino Mega. With the Arduino Mega, the team was able to use more than one port and properly wire two sensors onto the board. However, when running code for multiple sensors, the team encountered several errors. At first the sensors were not reading anything on the serial monitor. After some troubleshooting, a single sensor was capable of collecting data, while the other output unreliable readings. After further adjusting the code, both sensors were capable of collecting data at a much slower frequency. In addition, when a sensor was moved, the other would stop collecting data. Considering that the sensors were placed on a moving wheelchair, this error presented many issues.

The team believes that these issues are not due to the code or the Arduino board/type of Arduino board but are a result of the quality of the sensors. With more time and a larger budget, the team would have been able to experiment with different and more advanced distance sensors and would likely be able to run multiple sensors off a single Arduino board. This would have allowed the team to avoid many

obstacles encountered during the duration of the project and would have allowed the final product to better fit the specifications.

For the final design, it was decided to utilize multiple Arduino Uno boards instead of running multiple sensors on one Arduino Mega board. Each Arduino Uno board was reliably able to run code and collect accurate distance readings from one time-of-flight sensor. The final set up consisted of two sensors, one detecting close distances at 0.5 meters and the other detecting further distances at 1.5 meters, each on their own Arduino Uno board and connected to one breadboard. Both Arduinos were connected to the breadboard with the same set of three LEDs; one green, one yellow and one red. The far distance sensors controlled the green and yellow LEDs while the close distance sensors controlled the red LED. Our logic consisted of a simple IF statement that would control the LEDs depending on the distance collected by the sensors. See Appendix 1B for further detail of the written code. See Appendix 2I for depiction of the LED circuit.

Testing

Four sections of testing were conducted on the TFmini-S Lidar laser range sensors. These sections were static distance testing, static angle testing, static surface testing, and dynamic testing. Each section of testing was conducted with the purpose of defining a specification of the sensors necessary to create a system that meets the project specifications.

Static distance testing was conducted to test the measurement capabilities of the TFmini-S Lidar laser range sensors. This was done by evaluating the range of measurement in which the sensors can provide an accurate reading. Each sensor was placed at 12 marked distances away from a wall. The sensors were then placed on an evenly flat surface facing directly perpendicular to the wall. Three iterations of data were collected at each of these locations and the results were averaged and compared to the actual distances. See Appendix 2A for an image depicting the static distance testing set-up. From this testing it was determined that the sensors were capable of reading accurate distances within the range of 30 to 575 cm. It was also found that the time-of-flight sensors have a 2-degree field of view. This means that the sensors have a variability of two degrees in all directions. As a result, when the sensor is placed flat on a surface (with no elevation), the sensor will detect the surface it is sitting on. As a result, it was determined that the time-of-flight sensors need to be placed at a minimum elevation of 1 inch. This process also allowed the team to identify 2 unreliable sensors that did not output readings within 90% accuracy. See Appendix 3A for a graph depicting the static distance testing results.

Static angle testing was conducted to determine what angle (relative to the ground) the sensors are able to accurately detect a distance measurement. Each sensor was placed both 1 and 3 meters away from the object of interest. This object was rotated from 0 to 80 degrees and distance measurements were collected every 10 degrees. This process was repeated for 3 iterations on each sensor and results were averaged and compared to the actual distances. The test set-up is depicted in Appendix 2B. It was determined that the sensors were able to read up to 70 degrees relative to the ground with a 90% accuracy, with the optimal angle being 67 degrees. These results are further depicted in Appendix 3B. This information is crucial in determining how the team will mount the sensors on the wheelchair and the maximum distance measurement the sensors will be capable of accurately reading.

Static surface testing was performed to determine how the sensors will collect data on various surfaces that may typically be found in the environments that users will utilize their wheelchairs. The surfaces tested were artificial turf to imitate grass, a cement block to represent a sidewalk, a curved surface to represent a curb, and a reflective surface to model glassy/polished tile. Appendices 2B and 2C show the concrete and artificial turf tests respectively. Data was collected 1 meter away at angles between 0 to 80 degrees. Similarly to the static angle testing, each measurement was collected 3 times per sensor at 10 degree increments. Additionally, these data were averaged and compared to the actual distances. For the artificial turf and concrete it was found that the sensors were able to read up to 70 degrees relative to the floor with a 90% accuracy. However, the reflective surface did not display the same accuracy. The curved surface did not affect the sensors' accuracy and the proper distances were recorded independent of the distances along the curved surface. This testing gave confidence that the system could be implemented in real life scenarios like grass, concrete, and curbs. It was also determined that reflective surfaces can lead to inaccuracies in the sensors.

Dynamic testing was conducted to collect data that will accurately represent the system in full use. The goal of dynamic testing is to identify any final errors that may occur when the user is operating the system. This testing was conducted with the final set up, utilizing the mounts at the desired angles and heights, properly ventilated and housed wirings to all Arduinos and breadboards, and fully functioning code. To verify that the system was working properly, the further out sensor was connected to a computer to view the serial monitor and observe output distances. This setup can be seen in Appendix 2G. From this testing, it was determined that the system was able to measure expected distances with a minimum of 90% accuracy to detect and alert a user when approaching a drop off. Image 1 below depicts a single iteration of dynamic testing. The measured distance can be seen increasing from about 200 cm to 245 cm as the wheelchair approaches a drop off, then decreasing back to about 200 cm when the wheelchair was moved and no longer in range of the drop off.

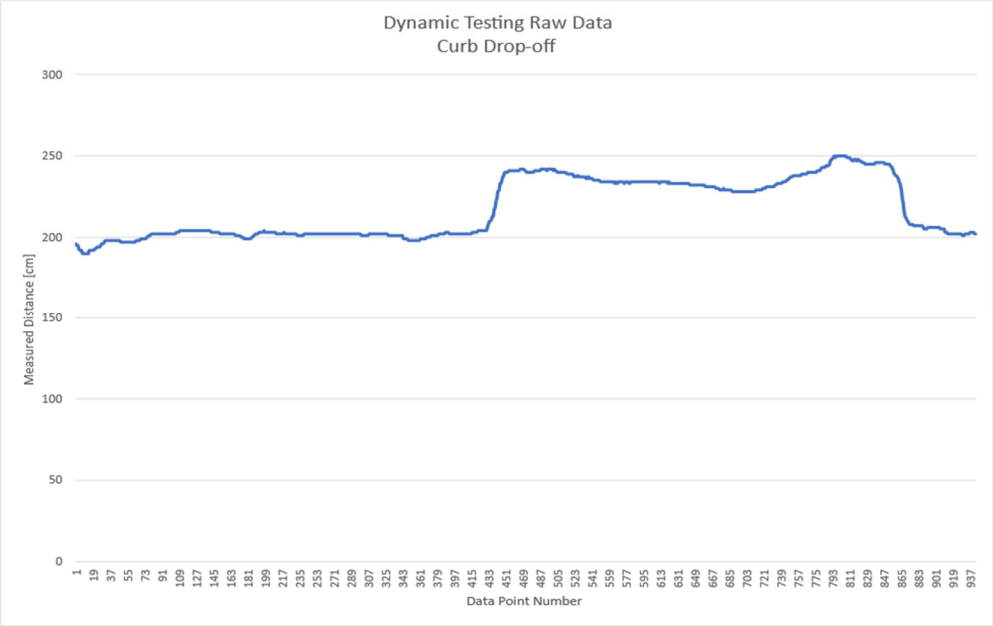
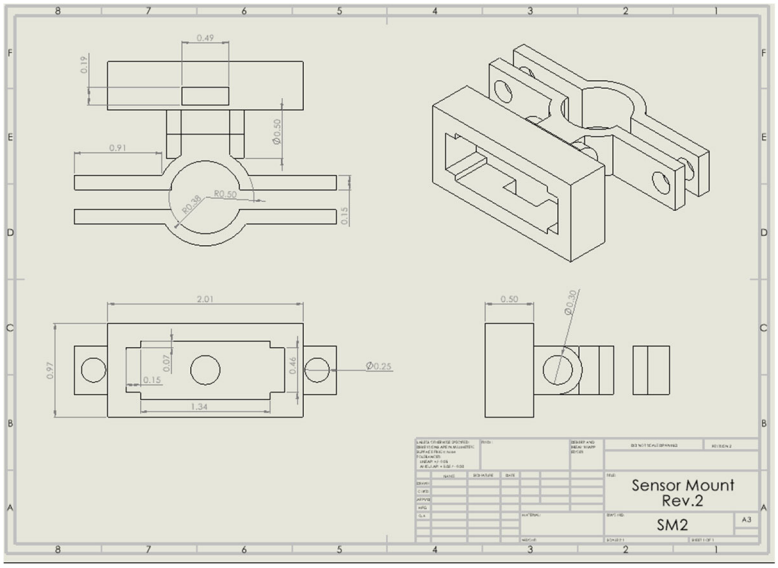


Image 1: A Trial of Dynamic Testing

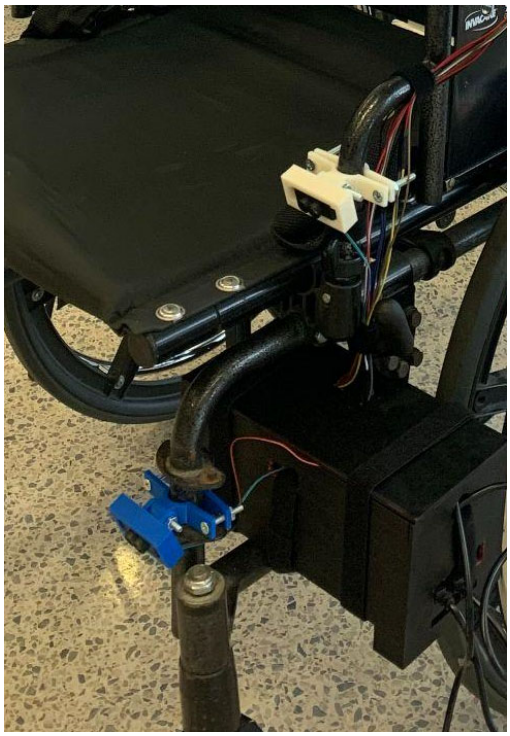
Design Process

Image 2: Drawing File of Sensor Housing



Mounts were designed for each of the two time-of-flight sensors using SolidWorks and printed using a PLA based 3D printer. When designing the mounts, the project's specifications, and limitations of the wheelchair and sensors were taken into account. One important consideration was that the system must not interfere with the function of the wheelchair. This specification was met by placing the sensor mounts along the outer frame of the wheelchair. It was also crucial that the sensor mounts remained secure along the wheelchair frame and didn't slide down as a result of the wheelchair's movement. Hinges were also designed into the sensor mounts to allow the team to change the sensor angles, and therefore, vary the distances of the close and far distance readings. The heights at which the sensors were placed was limited by the quantity of open space along the wheelchair frame. The time-of-flight sensor measuring further distances was placed 68.6 cm above the ground reading distances of 200 cm away from the wheelchair. As a result, the far reading sensor was placed at a 70-degree angle. The time-of-flight sensor measuring closer distances was mounted 39 cm above the ground, reading distances 100 cm away from the wheelchair. The sensor was, therefore, placed at a 67-degree angle. Both these angles are within the allowable range of 0 to 70 degrees meaning the distance reading will be 90% accurate. The time-of-flight sensor mount design and placements along the wheelchair can be seen in images 2 and 3 respectively as well as in Appendix 2D.

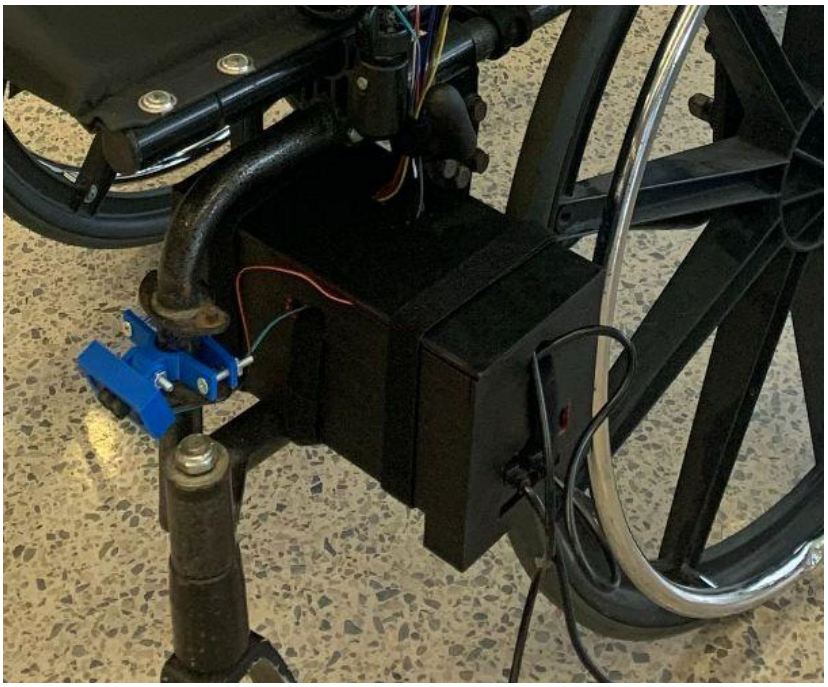
Image 3: Sensor Housing on Wheelchair



Each time-of-flight sensor was connected to an Arduino Uno board. The wiring connections can be seen in Appendix 2E. These Arduino Uno boards were powered by a wireless battery pack. The Arduino Uno boards and battery pack were placed in a box located below the wheelchair, to ensure that there were no interferences with the usage of the wheelchair. As a result, the placement of

this box allowed the user to fold the wheelchair for transportation purposes. This box was utilized from the previous team and modified with holes located in specific areas along its length to allow the wiring to be both consolidated and organized. An image depicting the box's location and design can be seen below in Image 4.

Image 4: Location of Box that houses Arduinos



The data collected from the time-of-flight sensors was relayed from the Arduino Unos to a breadboard which displayed a green, yellow, or red LED light to the user. This breadboard was placed in yet another box-like mount. It was critical that this mount was in the user's line of vision and was easily able to display the LED lights. Again, it was also made certain that this mount would not obstruct the usage of the wheelchair. To accommodate these specifications, it was determined that the breadboard mount would be placed next to the right-side arm rest of the wheelchair. It was decided that the mount would connect below the arm rest, then protrude laterally before extending upwards to lay parallel to the arm rest. This mount had three holes along the top face to allow the green, yellow, and red LED lights to protrude and be displayed to the user. The back face of the box-like mount remained open to allow the team to remove and insert the breadboard when necessary. The design and attachment of this mount can be seen in images 5 and 6 respectively.

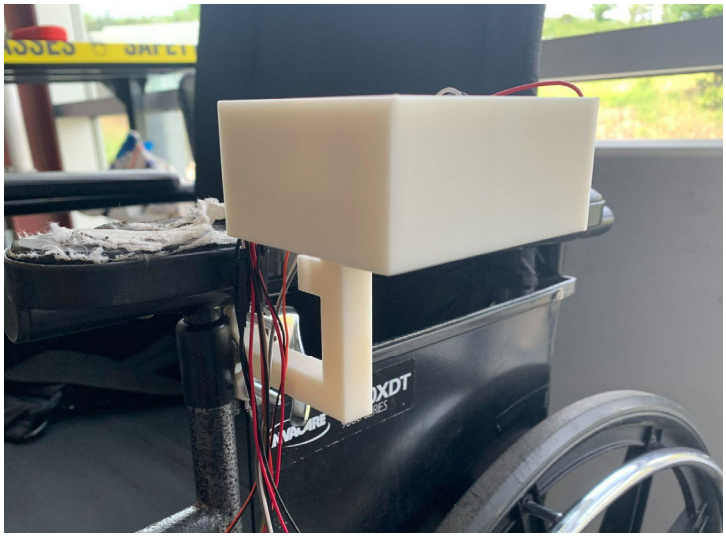


Image 5: Placement of Breadboard and LED Warning System

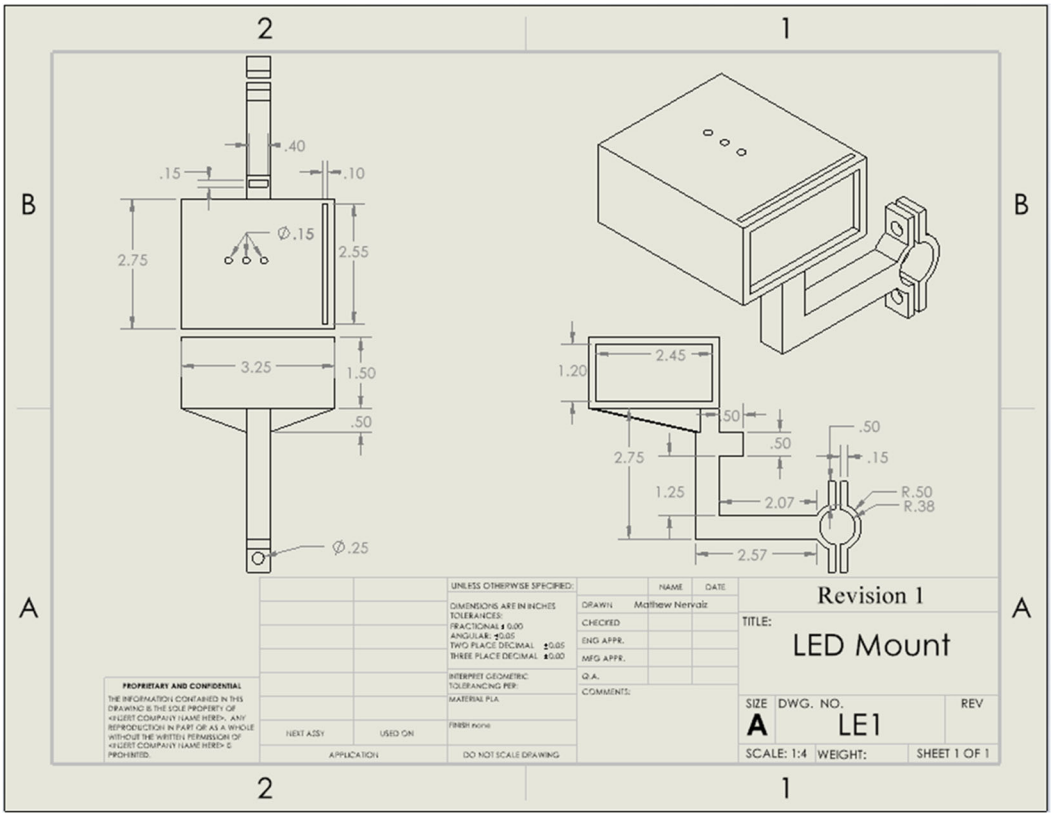


Image 6: Drawing File for Breadboard Mount

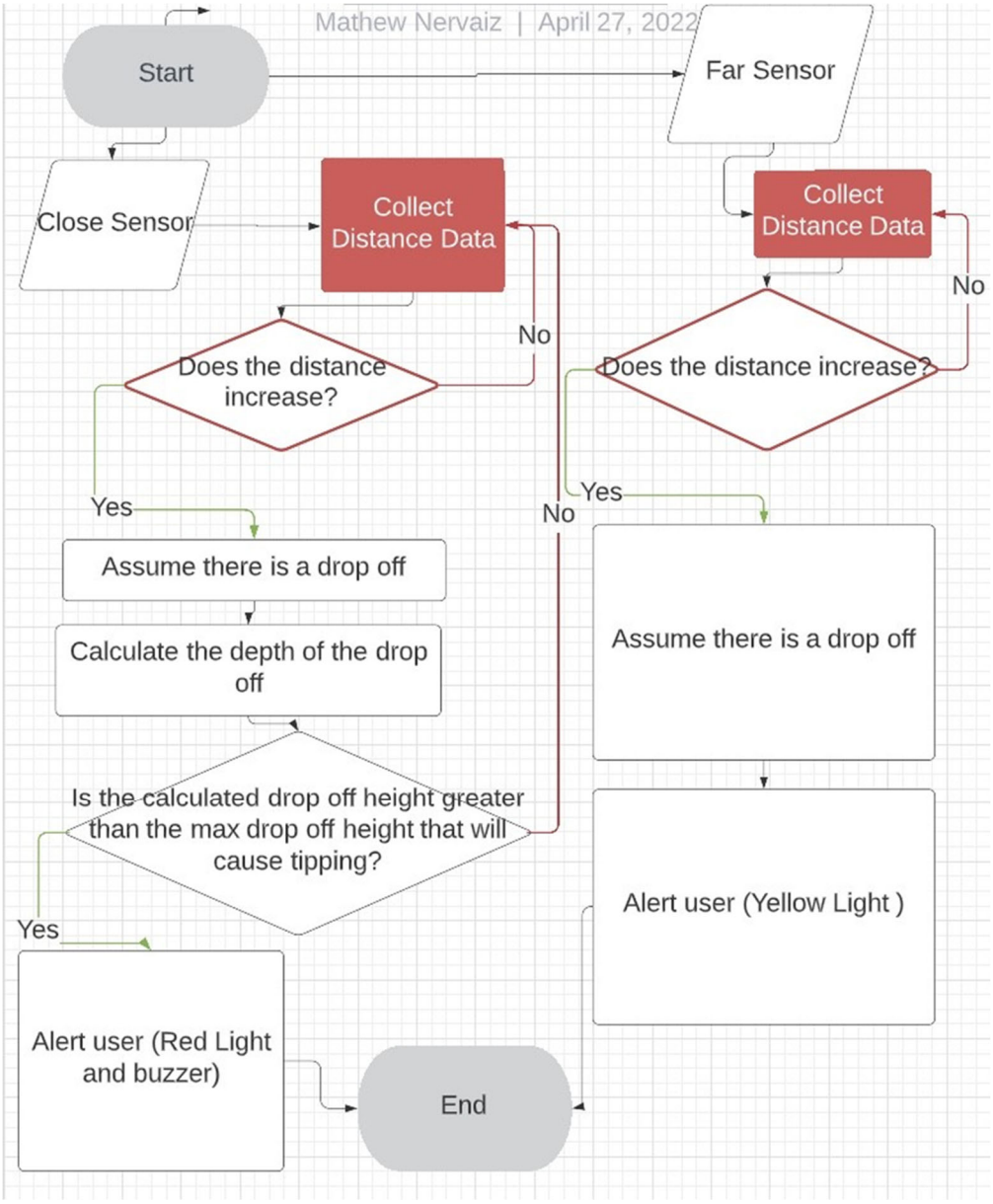
Conclusions

It was expected that the sensors would have many errors in readings that would need to be filtered out by the code and that this project would focus a significant amount of time on this issue. Through extensive testing, this turned out not to be true and in fact the sensors proved to be very accurate and dependable. Although, more testing could be done to conclude the significance of these results. Many of the tests that were completed had only three rounds to compare between. More rounds of testing could be completed which would bolster confidence in any results. While the sensors performed well when used alone, when trying to run more than one sensor or with any extended wires, the sensors became very undependable. For this reason, if this project were to move forward, it would be suggested to invest in better sensors or at the very least a more permanent wiring solution. There were many challenges faced throughout the course of this project due to faulty readings and inaccurate results from the sensors. Some of these were simply due to wiring but some were due to the quality of the sensors. With better sensors, it is believed that more accurate data could be collected, and the project would better fit specifications. Another potential solution is switching back to Raspberry Pi. To this team's knowledge, the issues that were faced were not shared with the Raspberry Pi system. It is believed that the Raspberry Pi system has greater capabilities than the Arduino Uno's and could be used to obtain more dependable results. Another thing that should be added onto this project is a buzzer. One of the most mentioned customer specifications was having an auditory warning as well as a visual warning and this is something that we were unable to get to due to time spent facing issues with the code and sensors. Although the sensors were accurate, they still gave out errors at times. An improvement to the system could be made by adding filtering to the code which would give a warning when the sensors got stuck on -1 or -3 as this issue came up often.

Appendix

1.) Flowcharts and Code

1A.) Final Version of the Flowchart



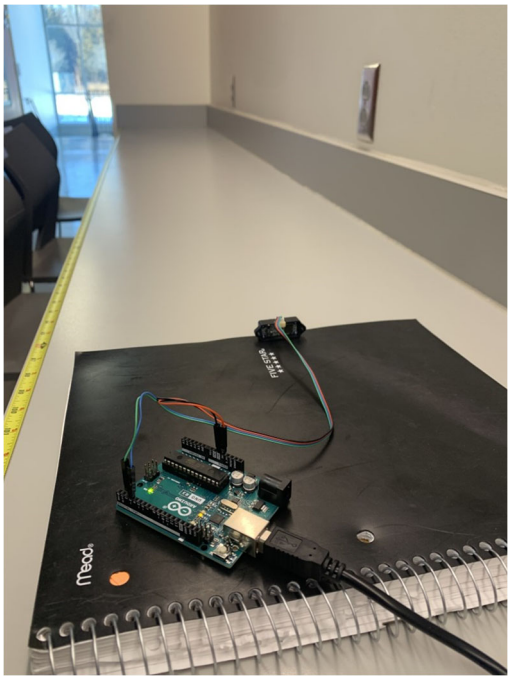
1B.) Final Version of Code

This code was used on the Uno's and includes the LED warnings.

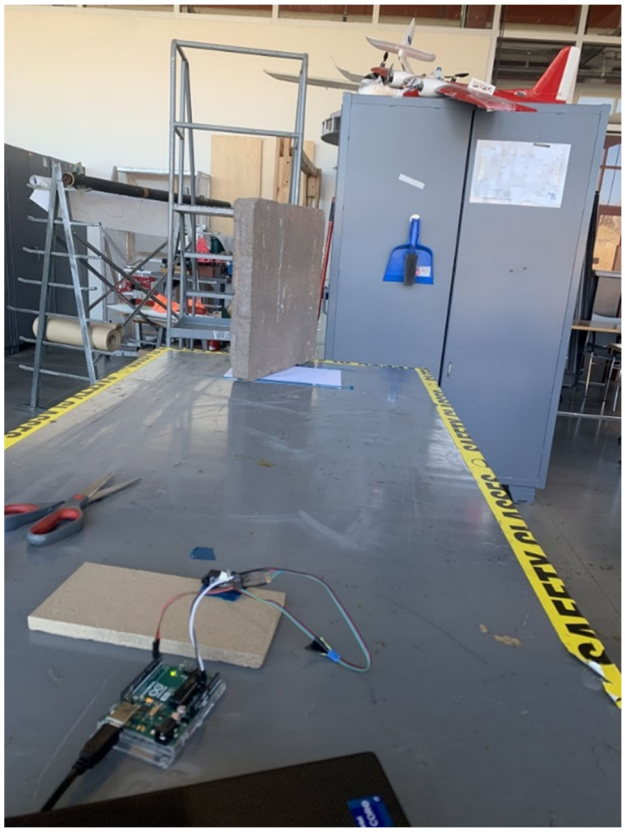
```
#include <SoftwareSerial.h>  //header file of software serial port

SoftwareSerial Serial1(2,3); //define software serial port name as Serial1 and define pin2 as RX and pin3 as TX

int dist; //actual distance measurements of LiDAR
int strength; //signal strength of LiDAR
int check;  //save check value
int i;
int uart[9];  //save data measured by LiDAR
const int HEADER=0x59;  //frame header of data package

const int LIGHT_SENSOR_PIN = A0; // Arduino pin connected to light sensor's pin

const int LED_RED = 4; // Arduino pin connected to LED's pin
const int LED_YELLOW = 6; // Arduino pin connected to LED's pin
const int LED_GREEN = 7; // Arduino pin connected to LED's pin

const int ANALOG_THRESHOLD = 500;


// variables will change

int analogValue;

void setup() {

  Serial.begin(9600); //set bit rate of serial port connecting Arduino with computer
  Serial1.begin(115200);  //set bit rate of serial port connecting LiDAR with Arduino

pinMode(LED_RED, OUTPUT); // set arduino pin to output mode
pinMode(LED_YELLOW, OUTPUT);
pinMode(LED_GREEN, OUTPUT);
}

void loop() {

  if (Serial1.available()) {  //check if serial port has data input
    if(Serial1.read() == HEADER) {  //assess data package frame header 0x59
      uart[0]=HEADER;
      if (Serial1.read() == HEADER) { //assess data package frame header 0x59
        uart[1] = HEADER;
        for (i = 2; i < 9; i++) { //save data in array
          uart[i] = Serial1.read();
        }
        check = uart[0] + uart[1] + uart[2] + uart[3] + uart[4] + uart[5] + uart[6] + uart[7];
        if (uart[8] == (check & 0xff)){ //verify the received data as per protocol
          dist = uart[2] + uart[3] * 256;     //calculate distance value
          strength = uart[4] + uart[5] * 256; //calculate signal strength value
          Serial.print("dist = ");
          Serial.print(dist); //output measure distance value of LiDAR
          Serial.print('\t');
          Serial.print("strength = ");
          Serial.print(strength); //output signal strength value
          Serial.print('\n');
//
if(dist < 200) {
//digitalWrite(LED_RED, LOW);
digitalWrite(LED_YELLOW, LOW);
digitalWrite(LED_GREEN, HIGH);
}
else if (dist >= 200){
////digitalWrite(LED_RED, LOW);
digitalWrite(LED_YELLOW, HIGH);
digitalWrite(LED_GREEN, LOW);
}
else if (dist <= 100){
digitalWrite(LED_RED, HIGH);
digitalWrite(LED_YELLOW, LOW);
digitalWrite(LED_GREEN, LOW);
}
//  FOR CLOSE SENSOR
 if(dist <= 100){
digitalWrite(LED_RED, HIGH);
digitalWrite(LED_YELLOW, LOW);
digitalWrite(LED_GREEN, LOW);
  }
else if(dist > 100){
   digitalWrite(LED_RED, LOW);
digitalWrite(LED_YELLOW, LOW);
digitalWrite(LED_GREEN, LOW);
}

      }
    }
  }
}
}
}
```

2.) Figures

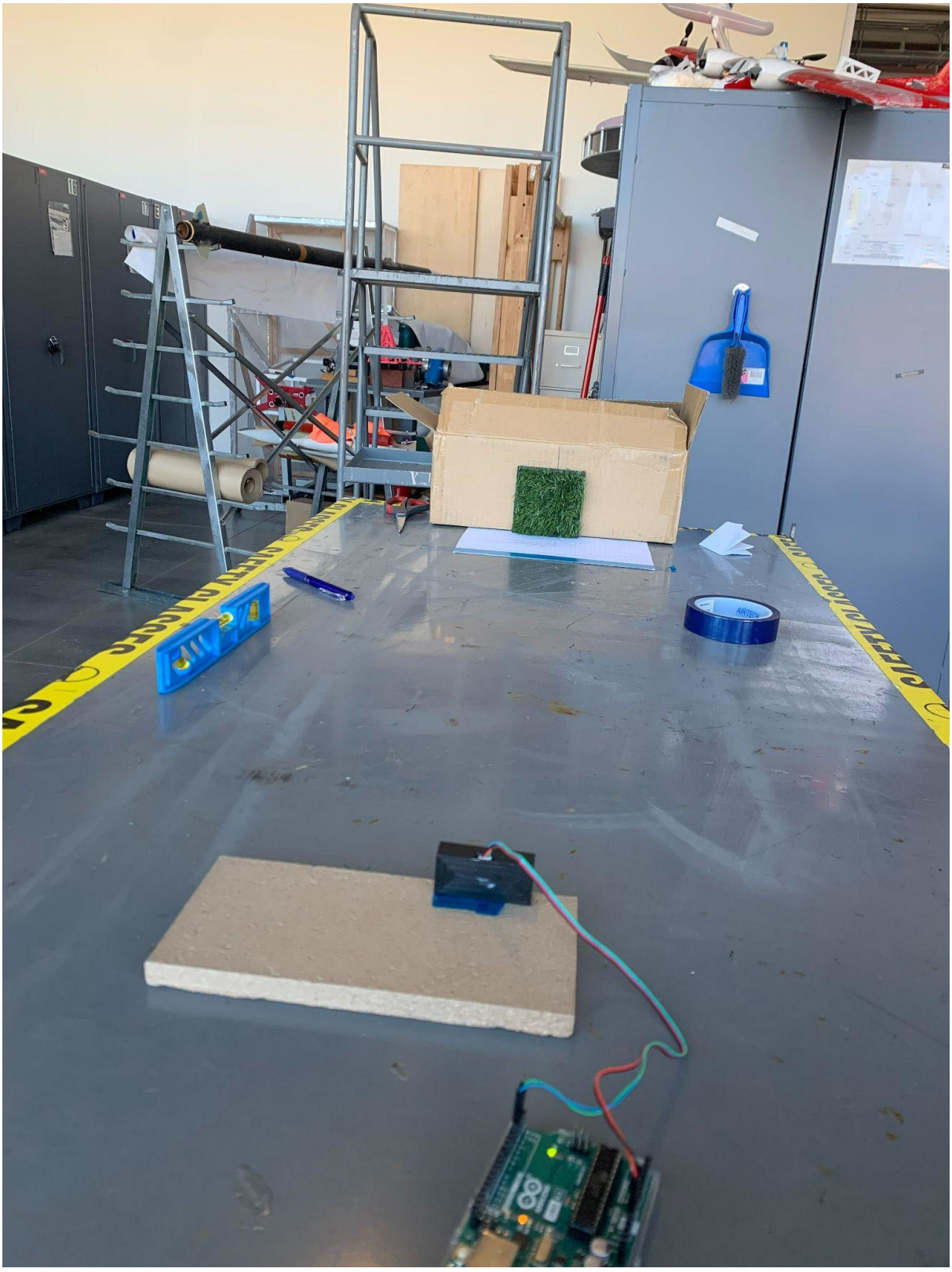
2A.) Static Testing Setup



2B.) Concrete Surface Angle Testing

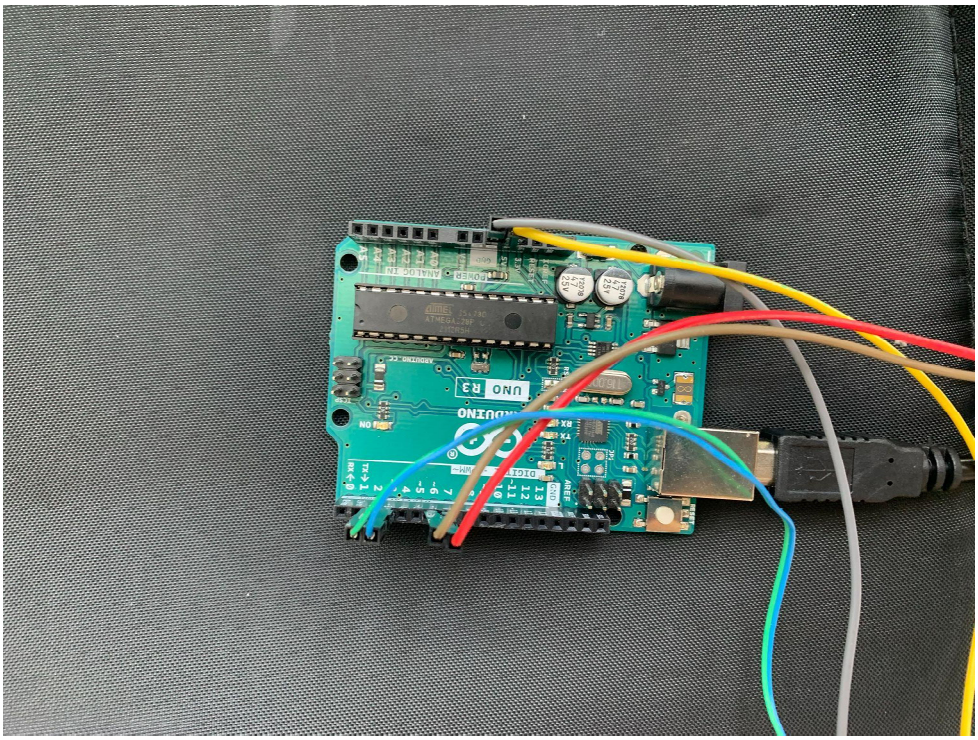


2C.) Artificial Turf Surface Testing



2D.) Housing Mount for Far Distance Sensor

2E.) Wiring of Arduino Board

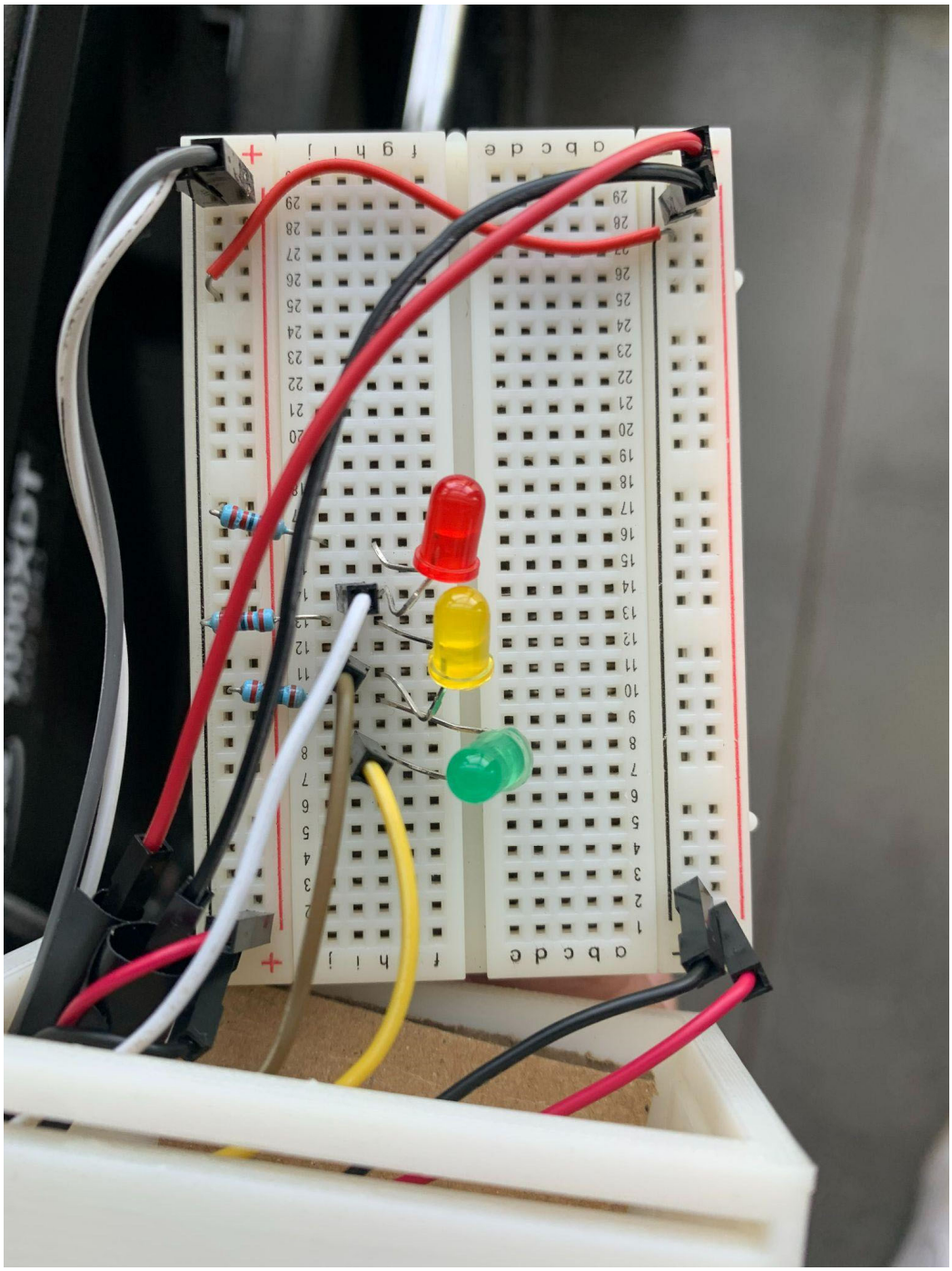


2F.) Poster Presentation

2G.) Dynamic Testing

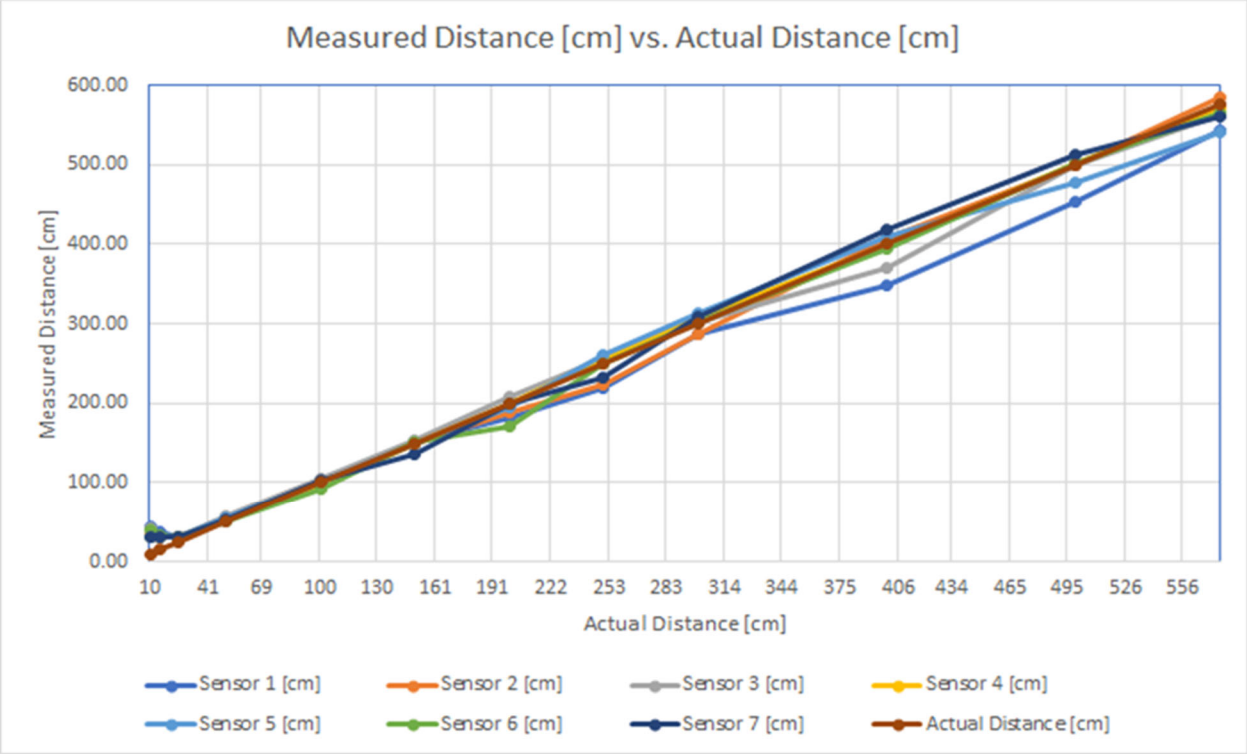
2H.) TFmini-S Lidar Laser Range Sensor

2I.) Breadboard with LED’s

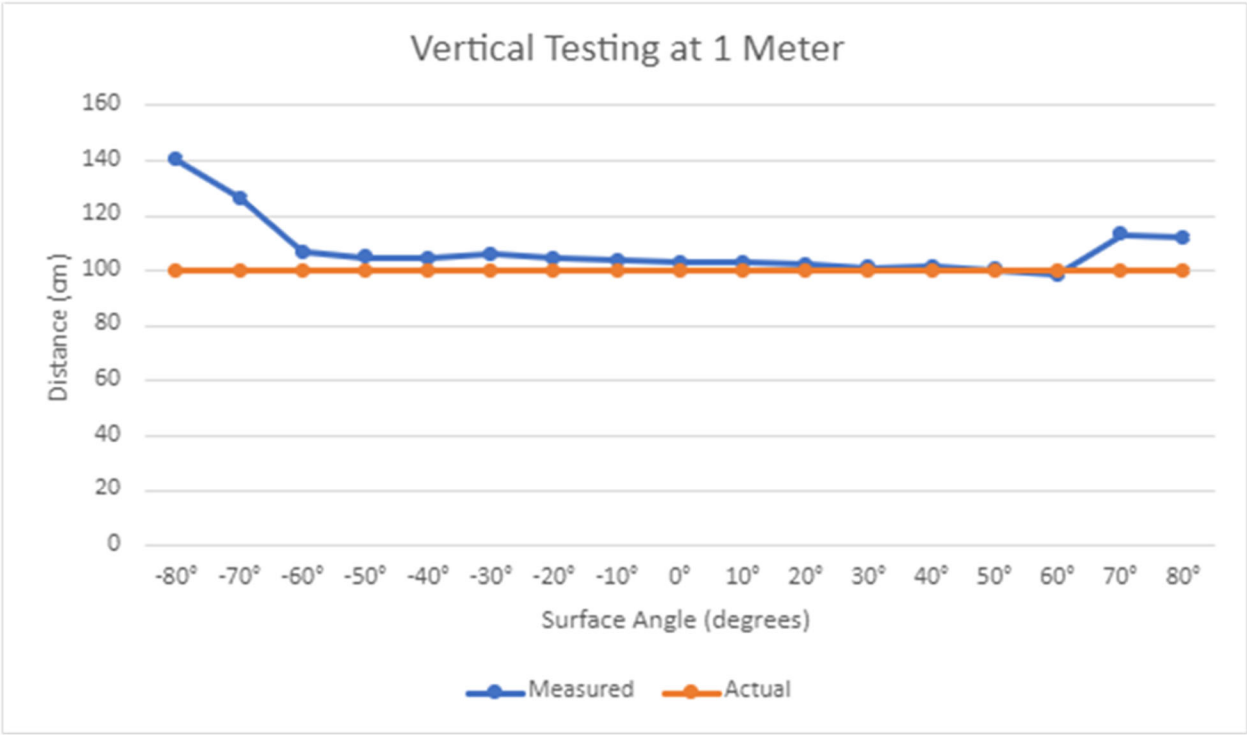


3.) Graphical Test Results

3A.) Static Testing Results



3B.) Angle Testing Results



References

[1]Xiang, H., Chany, A., & Smith, G. (2006, February). Wheelchair related injuries treated in US emergency departments. Retrieved December 06, 2021, from https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2563507/

[2] Robert, H. T. (2021). What is MS? Retrieved October 27, 2021, from https://www.nationalmssociety.org/What-is-MS/Definition-of-MS

[3] Gaal, R. P., & Rebholtz, N. (1999, January 1). Wheelchair rider injuries: Causes and consequences for wheelchair design and selection. Department of Veterans Affairs. Retrieved December 6, 2021, from https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.469.3230&rep=rep1&type=pdf.